Serveur-ansible

Commencer

Suivez ces étapes pour démarrer avec Ansible. Certaines de ces étapes supposent que vous travaillerez sur un hôte Ubuntu. Si vous utilisez un MAC ou une autre version de Linux, veuillez consulter la documentation officielle d'Ansible pour obtenir de l'aide sur l'installation d'Ansible. La plupart de ces étapes au-delà de l'installation seront les mêmes quelle que soit la version de Linux utilisée.

Préparer le commutateur pour Ansible

Créer un compte utilisateur

Si un compte utilisateur existe déjà sur le commutateur il peut être utilisé pour se connecter via SSH et apporter des modifications, il n'est pas nécessaire de créer un nouveau compte. S'il n'y a pas de compte déjà créé, voici un exemple de la façon d'en créer un :

```
#
local-user hp
 password simple hp123
 service-type ssh http https
 authorization-attribute user-role network-admin
#
line vty 0 15
 authentication-mode scheme
 user-role network-admin
```

Activer SSH

```
ssh server enable
ssh user hp service-type all authentication-type password
```

Activer NETCONF

```
netconf ssh server enable
```

Activer SCP

Cette étape n'est requise que lors de l'utilisation de modules Ansible spécifiques qui copie les fichiers de l'hôte de contrôle Ansible local vers le commutateur HP. Trois de ces modules incluent: `comware_install_config`, `comware_install_os`, et `comware_file_copy`.

```
scp server enable
```

Installer Ansible

Dans une session de terminal sur votre machine Linux ou putty, exécutez les commandes suivantes :

```
sudo apt-get install python-pip
```

```
sudo apt-get install python3-pip
```

```
$ sudo pip install markupsafe
$ sudo pip install ansible
```

Installer la bibliothèque Python HP Comware 7

Pendant une session de terminal sur votre machine Linux, exécutez l'un des blocs de commandes suivants :

Dernières de la source

```
$ git clone https://github.com/HPENetworking/pyhpecw7.git
$ cd pyhpecw7
$ sudo python setup.py install
```

Dernière version stable via PIP (pas encore pris en charge)

```
$ sudo pip install pyhpecw7
```

Modifier le nom de l'host

Vous allez entrer la commande vi /etc/hosts (vous pouvez installer *vim* pour modifier le fichier plus facilement). Une fois dans le fichier changer le nom de votre hosts, vous le reconnaitre grâce à son adresse IP. Puis vérifier la connexion grâce à commande Ping.

```
vi /etc/hosts

sudo ping sw1
```

Il se pourrait qu'il y est un problème de version en installent ansible donc fait la commande :
Pour revenir à une version antérieure. *sudo pip install --system bcrypt==3.1.7*

Vérifiez que la bibliothèque est correctement installée

Entrez une nouvelle session de terminal et entrez l'interpréteur Python.

```
python3
```

Importez l'objet périphérique HPCOM7.

```
>>> from pyhpecw7.comware import HPCOM7
```

Préparez les arguments nécessaires pour vous connecter à l'appareil. Ceux-ci incluent le nom d'hôte ou l'adresse IP du commutateur, le nom d'utilisateur et le mot de passe utilisés pour se connecter au commutateur (ceux-ci doivent être préconfigurés sur le commutateur) et un numéro de port facultatif. Par défaut, c'est **830.**

Cet exemple suppose que le nom d'hôte du commutateur est hp1.

```
>>> args = dict(host='hp1', username='hp', password='hp123', port=830)
>>>
>>> device = HPCOM7(**args)
>>>
```

Étant donné que cette bibliothèque utilise NETCONF, qui est orienté connexion, vérifiez que l'objet périphérique (HPCOM7) n'est pas encore connecté au commutateur**.**

```
>>> device.connected
False
```

Remarque : assurez-vous que vous pouvez envoyer un ping à votre commutateur et assurez-vous que NETCONF est activé !!!!

Établissez une connexion au commutateur, c'est-à-dire ouvrez une session NETCONF et vérifiez qu'elle est maintenant connectée au commutateur.

```
>>> device.open()
<ncclient.manager.Manager object at 0x7fb2a3a2aa50>
>>>
>>> device.connected
True
>>>
```

Vous avez maintenant vérifié avec succès que la bibliothèque a été installée et que vous pouvez vous connecter au commutateur.

Une étape facultative consiste à utiliser l'une des bibliothèques de fonctionnalités pour collecter des données de configuration telles que celles d'un VLAN.

Remarque : le VLAN 20 a été préconfiguré sur le commutateur pour cet exemple.

```
>>> from pyhpecw7.features.vlan import Vlan
>>>
>>> vlan = Vlan(device, '20')
>>>
>>>
>>> vlan.get_config()
{'name': 'DEMO_VLAN', 'vlanid': '20', 'descr': 'VLAN 0020'}
```

Enfin, fermez la connexion à l'appareil.

```
>> >  dispositif . fermer ()
```

Installer les modules HP Ansible

Retournez d'abord dans votre répertoire personnel.

```
$ cd
```

Effectuez un clone de ce projet.

```
$ git clone https://github.com/HPENetworking/ansible-hpe-cw7
```

Accédez au nouveau hp-ansible répertoire.

```
$ cd hp-ansible
```

Commencez à automatiser votre réseau HP

Introduction et exemple d'Ansible Playbook 1

Cette section couvrira une introduction de haut niveau à Ansible en examinant le premier exemple de playbook situé dans hp-ansible répertoire qui a également été téléchargé lorsque vous avez cloné le référentiel github.

Il existe quelques playbooks Ansible prédéfinis dans `hp-ansible` répertoire qui peuvent être examinés et exécutés pour apprendre à utiliser Ansible pour automatiser votre réseau HP. Cette section servira d'introduction à Ansible ainsi que de la façon de commencer à exécuter et à écrire vos propres playbooks.

Noter:
Tous les exercices ci-dessous supposent que votre shell se trouve dans `hp-ansible/` répertoire. Si ce n'est pas le cas, certains exercices peuvent ne pas fonctionner. De même, si vous souhaitez expérimenter avec des playbooks personnalisés, ils doivent résider dans `hp-ansible/` répertoire.
Ouvrez le fichier appelé `hp-vlans.yml`. Il s'agit d'un playbook qui automatise le provisionnement des VLAN sur les commutateurs HP.

Voici le contenu du fichier `hp-vlans.yml`

```
---

 - name: VLAN Automation with Ansible on HP Com7 Devices
   hosts: hp1
   gather_facts: no
   connection: local

   vars:
     username: hp
     password: hp123

   tasks:

     - name: ensure VLAN 10 exists
       comware_vlan: vlanid=10 name=VLAN10_WEB descr=LOCALSEGMENT username={{ username }} password={{ pass

     - name: ensure VLAN 20 exists
       comware_vlan: vlanid=20 name=VLAN20 state=present username={{ username }} password={{ password }} h

     - name: ensure VLAN 10 does not exist
       comware_vlan: vlanid=10 state=absent username={{ username }} password={{ password }} hostname={{ in
```

Voici quelques détails importants concernant ce dossier.

- Ce fichier est un playbook Ansible.
- Tous les playbooks utilisent YAML comme format de données structurées
- Tous les fichiers YAML commencent par trois tirets ou tirets, c'est-à-dire `---`
- La première section du fichier a quelques paramètres qui incluent `name`, `hosts`, `gather_facts` et `connection`. Cela signifie le début d'une pièce dans le playbook. Remarque : il peut y avoir plusieurs jeux dans un seul playbook, mais cela n'affiche qu'un seul jeu. Voici un aperçu de ces quatre paramètres :
    - `name`: c'est arbitraire et une description textuelle qui s'affiche sur le terminal lors de l'exécution du playbook

- hosts: cela signifie l'hôte ou le groupe d'hôtes sur lesquels les tâches d'automatisation seront exécutées. L'hôte ou le groupe d'hôtes sont définis dans le fichier d'inventaire Ansible (présenté ensuite)
- gather_facts: noet connection: local sont requis ici car ces modules n'utilisent pas le mécanisme de connexion Ansible par défaut pour communiquer avec les commutateurs, c'est-à-dire SSH. Ces modules utilisent NETCONF. Il est également possible de définir ces valeurs une seule fois pour un environnement donné dans l'ansible.cfg fichier.

Avant de détailler le playbook, le fichier d'inventaire Ansible est examiné. Le nom de ce fichier est arbitraire, mais le fichier dans hp-ansible répertoire s'appelle hosts. Voici le fichier d'exemple, que vous pouvez également voir si vous l'ouvrez dans un éditeur de texte.

Avant de détailler le playbook, le fichier d'inventaire Ansible est examiné. Le nom de ce fichier est arbitraire, mais le fichier dans hp-ansible répertoire s'appelle hosts. Voici le fichier d'exemple, que vous pouvez également voir si vous l'ouvrez dans un éditeur de texte.

```
[all:vars]
username=hp
password=hp123

[switches]
hp1
hp2
```

Le premier bloc contient des variables qui peuvent être utilisées par tous les appareils d'un playbook. Certains des exemples et des playbooks de test extraient le nom d'hôte et le mot de passe du fichier d'inventaire tandis que d'autres les ont définis dans le playbook. Ceci est fait pour montrer quelques-unes des options disponibles pour l'utilisateur lors de l'utilisation d'Ansible.
Le deuxième bloc représente un groupe et deux hôtes. Pour cet exemple, les noms des hôtes correspondent à ce qui est dans /etc/hosts fichier. Si DNS n'a pas été configuré du tout, il est possible d'entrer des adresses IP ici. Plusieurs groupes peuvent être configurés dans l'inventaire, tout comme les variables pour chaque groupe ou hôte particulier.

Remarque : assurez-vous d'utiliser vos noms d'hôtes et adresses IP corrects si vous ne les avez pas utilisés hp1 et hp2 dans votre fichier /etc/hosts.

A présent, vous devriez avoir une bonne compréhension du *début* du playbook et jouer comme indiqué dans hp-vlans.yml. L'étape suivante consiste à revoir les tâches décrites dans la pièce. Vars section sera examinée ci-dessous.

Chaque tâche est censée être facilement interprétée. Tout comme le début du jeu utilise name, il en va de même pour chaque tâche (bien que ce ne soit pas une exigence). Name est du texte brut qui s'affiche sur le terminal pendant l'exécution de la tâche. Les tâches de cette pièce utilisent toutes le module appelé comware_vlan. C'est là que les modules entrent en action. Les modules sont appelés et envoyés en paramètres comme indiqué. Voici quelques-uns de ces paramètres pour le module vlan : vlanid, name, et descr. Ces valeurs devraient être explicites, mais elles servent à configurer un VLAN avec ces attributs respectifs.

Les autres paramètres tels que hostname, username, password sont requis pour tous les modules comware Ansible. Comme vous pouvez le voir, il y a des accolades autour des valeurs attribuées à ces paramètres. Ceux-ci signifient qu'ils sont des variables. Vous devriez pouvoir voir deux variables qui ont été déclarées avant les tâches dans vars section.

Ils sont username et password et vous pouvez voir qu'ils sont référencés comme tels username={{ username }}. Il se trouve que dans cet exemple le paramètre du module porte le même nom que la variable. Comme indiqué ci-dessus, des variables comme celle-ci pourraient être déclarées dans l'inventaire, dans un playbook et à partir de fichiers externes (non traités ici).

Le seul cas particulier ici est inventory_hostname. Il s'agit d'une variable Ansible intégrée qui fait référence au nom d'hôte de l'hôte à partir du fichier d'inventaire Ansible. Étant donné que le hosts est défini sur hp1 pour cette lecture, inventory_hostname sera égal à hp1. Si, en haut du playbook, nous utilisions switches comme groupe d'hôtes automatisé, il inventory_hostname serait défini hp1 sur la première exécution et hp2 la deuxième exécution de la tâche.

Remarque : il n'est pas recommandé de stocker les informations d'identification dans un playbook comme celui-ci. Ceci est fait à des fins de démonstration. Une meilleure approche serait de les stocker dans le fichier hosts, le fichier de variables, le fichier caché ou d'utiliser le coffre-fort Ansible.

Encore un détail sur ce module :

- Le module VLAN est idempotent comme la plupart des modules. Le module VLAN garantit que le VLAN est dans le bon état. Cela signifie que si le VLAN est déjà dans l'état souhaité (créé, nom du VLAN, etc.), aucune modification n'est apportée à l'appareil
  Il est temps que vous exécutiez ce livre de jeu.

Dans le répertoire actuel où hp-vlans.yml se trouve le, exécutez la commande suivante :

```
$ ansible-playbook -i hosts hp-vlans.yml
```

Vous verrez alors la sortie suivante :

```
PLAY [VLAN Automation with Ansible on HP Com7 Devices] ***********************

TASK: [ensure VLAN 10 exists] *************************************************
changed: [hp1]

TASK: [ensure VLAN 20 exists] *************************************************
changed: [hp1]

TASK: [ensure VLAN 10 does not exist] *****************************************
changed: [hp1]

PLAY RECAP ********************************************************************
hp1                        : ok=3    changed=3    unreachable=0    failed=0
```

Remarque : -i est utilisé pour référencer le fichier d'inventaire que vous souhaitez utiliser pour le playbook. Il n'est pas nécessaire de toujours utiliser le –i drapeau. Une autre option consiste à définir ANSIBLE_HOSTS variable d'environnement. Cela peut être fait comme ceci:

```
$ export ANSIBLE_HOSTS=hosts
```

Exécutez à nouveau le même playbook. Vous verrez maintenant cette sortie :

```
PLAY [VLAN Automation with Ansible on HP Com7 Devices] ***********************

TASK: [ensure VLAN 10 exists] *************************************************
changed: [hp1]

TASK: [ensure VLAN 20 exists] *************************************************
ok: [hp1]

TASK: [ensure VLAN 10 does not exist] *****************************************
changed: [hp1]

PLAY RECAP ********************************************************************
hp1                        : ok=3    changed=2    unreachable=0    failed=0
```

Remarquez comment le VLAN 10 a été à nouveau modifié. C'est parce que nous l'avons supprimé dans notre dernière tâche avec state=absent et l'avons rajouté à nouveau dans la première tâche de la deuxième exécution. Étant donné que nous nous assurons toujours que le VLAN 20 existe sur le commutateur et qu'il existe déjà, il n'y a eu AUCUN changement sur le commutateur et vous auriez dû remarquer que la sortie du terminal est restée verte pour le VLAN 20.

Exemple Playbook 2

Dans cet exemple suivant, nous examinerons les attributs physiques des interfaces. Deux qui seront affichés sont le type d'interface, c'est-à-dire la couche 2 ou la couche 3 et la description de l'interface.

Ce playbook est le `hp-interfaces.yml` fichier.

```
---

  - name: Example for Configuration Interfaces
    hosts: hp1
    gather_facts: no
    connection: local

    vars:
      username: hp
      password: hp123

    tasks:

      - name: ensure these interfaces are routed
        comware_interface: name={{ item }} type=routed username={{ username }} password={{ password }} host
        with_items:
          - FortyGigE1/0/1
          - FortyGigE1/0/2
          - FortyGigE1/0/3
          - FortyGigE1/0/4

      - name: ensure the description for 1/0/1 is set
        comware_interface: name=FortyGigE1/0/1 description='ANSIBLE CONFIGURED THIS' username={{ username }

      - name: ensure these interfaces are bridged
        comware_interface: name={{ item }} type=bridged username={{ username }} password={{ password }} hos
        with_items:
          - FortyGigE1/0/9
          - FortyGigE1/0/10
```

Comme on peut le voir, le playbook commencera presque toujours avec les mêmes paramètres. Les tâches de cet exemple utilisent le `comware_interface` module.

Lors de la définition du type de port sur `bridged` ou `routed` aucun autre paramètre de configuration (comme la description) à l'exception de ceux indiqués ne peut être utilisé.

Comme vous devriez être en mesure de le constater, ce livre de jeu garantit que :

- FortyGigE1/0/1 - FortyGigE1/0/4 seront des ports routés
- La description de "ANSIBLE CONFIGURED" sera sur FortyGigE1/0/1
- FortyGigE1/0/9 - FortyGigE1/0/10 seront des interfaces pontées

Exécutez ce playbook en entrant la commande suivante :

```
$ ansible-playbook -i hosts hp-interfaces.yml
```

En fonction de votre système (ces noms d'interface existent sur le HP 5930(pour HP 5130 il faut tapez *GigabitEthernet 1/0/xx*)), vous devrez peut-être modifier les noms utilisés pour qu'ils soient compatibles avec les interfaces de votre périphérique Comware 7 spécifique. Vous pouvez également voir une sortie différente en fonction de l'état actuel du système.

Voici le résultat après avoir exécuté ce playbook :

```
PLAY [Example for Configuration Interfaces] ***********************************

TASK: [ensure these interfaces are routed] ************************************
changed: [hp1] => (item=FortyGigE1/0/1)
changed: [hp1] => (item=FortyGigE1/0/2)
changed: [hp1] => (item=FortyGigE1/0/3)
changed: [hp1] => (item=FortyGigE1/0/4)

TASK: [ensure the description for 1/0/1 is set] *******************************
changed: [hp1]

TASK: [ensure these interfaces are bridged]  **********************************
ok: [hp1] => (item=FortyGigE1/0/9)
ok: [hp1] => (item=FortyGigE1/0/10)

PLAY RECAP ********************************************************************
hp1                        : ok=3    changed=2    unreachable=0    failed=0
```

Remarquez comment les quatre premières interfaces ont été modifiées, une description a été configurée, mais les deux dernières interfaces n'ont pas été modifiées. C'est parce que FortyGigE1/0/9 et FortyGigE1/0/10 étaient déjà en mode ponté.

Exécutez à nouveau le playbook et il devrait correspondre à ceci :

```
PLAY [Example for Configuration Interfaces] ***********************************

TASK: [ensure these interfaces are routed] ************************************
ok: [hp1] => (item=FortyGigE1/0/1)
ok: [hp1] => (item=FortyGigE1/0/2)
ok: [hp1] => (item=FortyGigE1/0/3)
ok: [hp1] => (item=FortyGigE1/0/4)

TASK: [ensure the description for 1/0/1 is set] *******************************
ok: [hp1]

TASK: [ensure these interfaces are bridged]  **********************************
ok: [hp1] => (item=FortyGigE1/0/9)
ok: [hp1] => (item=FortyGigE1/0/10)

PLAY RECAP ********************************************************************
hp1                        : ok=3    changed=0    unreachable=0    failed=0
```

Notez maintenant qu'aucune modification n'a été apportée puisque le commutateur était déjà dans l'état souhaité pour chaque tâche.

Exemple Playbook 3

L'exemple suivant montre comment créer un canal de port ponté (groupe d'agrégation).

(Changer FortyGigE1/0/1 ➔ GigabitEthernet 1/0/x pour les Switch HP5130)

```
---

  - name: Example for Configuration of Portchannels
    hosts: hp1
    gather_facts: no
    connection: local

    vars:
      username: hp
      password: hp123

    tasks:

      - name: ensure these interfaces are bridged
        comware_interface: name={{ item }} type=bridged username={{ username }} password={{ password }} hos
        with_items:
          - FortyGigE1/0/1
          - FortyGigE1/0/2
          - FortyGigE1/0/3
          - FortyGigE1/0/4

      - comware_portchannel:
          group: 100
          members:
            - FortyGigE1/0/1
            - FortyGigE1/0/2
            - FortyGigE1/0/3
            - FortyGigE1/0/4
          type: bridged
          mode: dynamic
          lacp_mode: active
          min_ports: 2
          max_ports: 4
          username: "{{ username }}"
          password: "{{ password }}"
          hostname: "{{ inventory_hostname }}"
          state: present
```

Avant de configurer un canal de port, la première étape doit toujours être de s'assurer que les interfaces sont dans le bon mode. Puisque nous voulons configurer un groupe d'agrégation de ponts, nous devons nous assurer que les interfaces sont de type ponté. Cela se fait à l'aide du `comware_interface` module comme on vient de le montrer dans l'exemple précédent.

La tâche suivante utilise le `comware_portchannel` module pour s'assurer que le groupe d'agrégation pontée 100 existe, qu'il compte 4 membres, qu'il est dynamique et que le mode est défini sur dynamique. Les ports min et max sont également configurés.

Notez qu'une syntaxe différente a été utilisée pour cette tâche. Cela utilise la syntaxe YAML, qui utilise un retrait de 2 espaces sous le nom du module avec le paramètre et la valeur séparés par deux points. La raison pour laquelle cela est utilisé est que ce format est requis lors de l'envoi

de types de données *complexes* tels qu'une liste en tant que valeur d'un paramètre dans un playbook, et comme on peut le voir dans l'exemple ci-dessus, une liste est envoyée en tant que valeur pour `members`. L'autre chose à réaliser est que lorsque les variables sont transmises en tant que valeurs, elles doivent être entourées de guillemets comme indiqué pour `username`, `password`, et `hostname`.

Voici le résultat de l'exécution du `hp-portchannels.yml` playbook.

```
$ ansible-playbook -i hosts hp-portchannel.yml

PLAY [Example for Configuration of Portchannels] ****************************

TASK: [ensure these interfaces are bridged]  ********************************
changed: [hp1] => (item=FortyGigE1/0/1)
changed: [hp1] => (item=FortyGigE1/0/2)
changed: [hp1] => (item=FortyGigE1/0/3)
changed: [hp1] => (item=FortyGigE1/0/4)

TASK: [comware_portchannel ] ************************************************
changed: [hp1]

PLAY RECAP ******************************************************************
hp1                        : ok=2    changed=2    unreachable=0    failed=0
```

Exemple Playbook 4

L'exemple suivant montre comment appliquer une nouvelle configuration en temps réel pour être la nouvelle configuration en cours d'exécution du commutateur HP Comware7. Sous les couvercles, il utilise la fonction de restauration du commutateur.

Remarque : l'exemple donné utilise un fichier de configuration pour le commutateur HP 5930.

Remarque : assurez-vous que les informations d'identification que vous utilisez pour vous connecter au commutateur ne sont pas supprimées. Il en va de même pour s'assurer que NETCONF reste activer et que l'adresse IP de gestion reste la même.

AVERTISSEMENT AVERTISSEMENT AVERTISSEMENT

N'APPLIQUEZ PAS LA CONFIG DE DÉMO À VOTRE COMMUTATEUR OU VOUS PERDREZ LA CONNECTIVITÉ SANS APPORTER LES CHANGEMENTS REQUIS !

Le fichier de configuration qui sera appliqué se trouve à l'adresse `configs/demo.cfg`

Le fichier DOIT avoir les informations d'identification, la route par défaut, l'adresse IP de gestion, etc. requises pour maintenir la connectivité avec l'appareil. Le fichier de démonstration a TOUS les ports définis en mode ponté, aucun canal de port configuré et aucun VLAN configuré. C'est une configuration de switch *propre*. Il modifie également le nom d'hôte de l'appareil.

NE POUSSER PAS CE FICHIER SUR VOTRE APPAREIL SANS CHANGER LES CREDS, L'ADRESSE IP MGMT ET L'ITINÉRAIRE EN CONSÉQUENCE.

Voici le contenu du fichier :

```
#
 version 7.1.045, ESS 2415
#
 sysname HP5930_DEMO_1
#
 clock protocol none
#
 irf mac-address persistent timer
 irf auto-update enable
 undo irf link-delay
 irf member 1 priority 1
 irf mode normal
#
 lldp global enable
#
 system-working-mode standard
 fan prefer-direction slot 1 port-to-power
 password-recovery enable
```

Le playbook qui poussera ce fichier de configuration à basculer s'appelle `hp-install_config.yml`

Cela ressemble à ceci :

```
---

  - name: Example for using install config
    hosts: hp1
    gather_facts: no
    connection: local

    vars:
      username: hp
      password: hp123


    tasks:

      - name: install config file that will be the new running config
        comware_install_config:
          config_file='./configs/demo.cfg'
          diff_file='./configs/demo-diff.diff'
          commit_changes=true
          username={{ username }}
          password={{ password }}
          hostname={{ inventory_hostname }}
```

Pour installer la config, le comware_install_config module est utilisé. Le config_file paramètre est obligatoire et correspond au chemin absolu complet du fichier de configuration que vous souhaitez transmettre à l'appareil. Le diff_file est facultatif, mais affichera les fichiers de configuration existants et nouveaux, et commit_changes est requis comme protection. S'il n'est pas défini sur true, la configuration ne sera pas appliquée à l'appareil.

Exécutons ce playbook (n'oubliez pas de vous assurer que vos informations d'identification, IP et route sont correctement dans VOTRE fichier de configuration).

Utilisez la commande suivante pour exécuter le playbook :

```
$ ansible-playbook -i hosts hp-install_config.yml -v
```

Cette fois, nous avons utilisé le –v ou verbose flag qui montre un peu plus de détails sur ce qui se passe.

Voici la réponse :

```
PLAY [Example for using install config] ************************************

TASK: [install config file that will be the new running config] ***************
changed: [hp1] => {"active_files": {"backup": "flash:/safety_file.cfg", "config_applied": "flash:/demo.cfg

PLAY RECAP ********************************************************************
hp1                        : ok=1    changed=1    unreachable=0    failed=0
```

Ce module fait trois choses :

1. Enregistre la configuration de travail actuelle en tant que flash :/safety_file.cfg
2. Charge la configuration que vous avez transmise en tant que nouvelle configuration en cours d'exécution
3. Enregistre la nouvelle configuration en cours sur flash :/startup.cfg

Cela couvre quelques types d'exemples différents utilisant les modules pour automatiser les périphériques HP Comware 7. De plus, il existe un certain nombre de playbooks de test qui peuvent être trouvés dans le test-pbs répertoire. Il y a un playbook de test par fonctionnalité ou module. Chacun de ces playbooks contient de nombreux exemples d'utilisation d'un ou plusieurs modules, mais est surtout utilisé pour prouver que les modules fonctionnent correctement.

Utiliser ansible-doc

ansible-doc est un utilitaire qui offre aux utilisateurs des documents intégrés de type « page de manuel » pour les modules Ansible. –M indicateur qui l'accompagnent spécifient le répertoire dans lequel résident les modules.

```
$ ansible-doc -M library/ comware_vlan
```

Vous verrez la sortie suivante :

```
$ ansible-doc comware_vlan
> COMWARE_VLAN

  Manage VLAN resources and attributes for Comware v7 devices

Options (= is mandatory):

- descr
        Description for the VLAN (Choices: ) [Default: None]

= hostname
        IP Address or hostname of the Comware v7 device that has
        NETCONF enabled (Choices: ) [Default: None]
```

Lorsque vous commencez à utiliser des modules, il s'agit d'un excellent utilitaire de ligne de commande à référencer pour comprendre les paramètres pris en charge par chaque module. Il peut être utilisé non seulement pour les modules HP, mais également pour tout module Ansible natif.